Arabic Language WEKA-Based Dialect Classifier for Arabic Automatic Speech Recognition Transcripts

Areej Alshutayri , Eric Atwell , AbdulRahman AlOsaimy
James Dickins, **Michael Ingleby** and **Janet Watson**
University of Leeds, LS2 9JT, UK
ml14aooa@leeds.ac.uk, E.S.Atwell@leeds.ac.uk ,
scama@leeds.ac.uk, J.Dickins@leeds.ac.uk,
inglebymeltham@binternet.com, J.C.E.Watson@leeds.ac.uk

Abstract

This paper describes an Arabic dialect identification system which we developed for the Discriminating Similar Languages (DSL) 2016 shared task. We classified Arabic dialects by using Waikato Environment for Knowledge Analysis (WEKA) data analytic tool which contains many alternative filters and classifiers for machine learning. We experimented with several classifiers and the best accuracy was achieved using the Sequential Minimal Optimization (SMO) algorithm for training and testing process set to three different feature-sets for each testing process. Our approach achieved an accuracy equal to 42.85% which is considerably worse in comparison to the evaluation scores on the training set of 80-90% and with training set 60:40 percentage split which achieved accuracy around 50%. We observed that Buckwalter transcripts from the Saarland Automatic Speech Recognition (ASR) system are given without short vowels, though the Buckwalter system has notation for these. We elaborate such observations, describe our methods and analyse the training dataset.

1 Introduction

Language Identification or Dialect Identification is the task of identifying the language or dialect of a written text. The task of Arabic dialect identification may require both computer scientists and Arabic linguistics experts. The Arabic language is one of the worlds major languages, and there is a common standard written form used worldwide, Modern Standard Arabic (MSA). MSA is based on the text of the Quran, the holy book of Islam; and MSA is taught in Arab schools, and promoted by Arab civil as well as religious authorities and governments. There are many dialects spoken around the Arab World; Arabic dialectologists have studied hundreds of local variations, but generally agree these cluster into five main regional dialects: Iraqi Dialect (IRQ), Levantine Dialect (LAV), Egyptian Dialect (EGY), North Africa Dialect (NOR), and Gulf Dialect (GLF) which is a subclass of Peninsular Arabic. Studies in Arabic dialectology focus on phonetic variation (Alorifi, 2008; Biadsy et al., 2009; Horesh and Cotter, 2016; Sadat et al., 2014). Horesh and Cotter (Horesh and Cotter, 2016) confirmed that past and current research is focussed on phonetic and phonological variation between Arabic dialects: all examples that they presented are of phoneme variation, and they did not mention any work on text, or corpus-based research, or of lexical or morpho-syntactic or grammar variation. However, Arabic spoken dialect does include local words, phrases, and even local variant morphology and grammar. With the spread of informal writing, for example on social networks and in local-dialect blogs, news and other online sources, Arabs are starting to write in their dialects. Because of the dominance of the MSA standard, there are no official writing standards for Arabic dialects, so spelling, morphology, lexis and grammar can be subject to individual transcription choice: it is up to a dialect speaker to decide how to write down their text. Dialect speakers have been taught from school to write down everything in MSA, so they may well normalise or translate into MSA rather than phonetically transcribe words and utterances. Pronunciation of vowels in words constitute one of the key differences between Arabic dialects; but in written MSA, most vowels are omitted, leaving few clues to distinguish the source dialect.

Proceedings of the Third Workshop on NLP for Similar Languages, Varieties and Dialects,
pages 204–211, Osaka, Japan, December 12 2016.

All this makes it challenging to collect an Arabic dialects texts corpus. Previous DSL shared tasks (Zampieri et al., 2015) were based on officially recognised and differentiated languages (Bosnian v Croatian v Serbian, Malay v Indonesian etc.) with readily-available published sources: Each example is a short text excerpt of 20100 tokens, sampled from journalistic texts . Local and national Arabic news sources and other journalistic text may include some local words but are still permeated and dominated by MSA, so a DSL Arabic dialects journalisic text data-set would be contaminated with MSA/dialect code-switching, and blocks of MSA. The DSL organisers tried instead to gather dialect data more directly from dialect speakers, and tried to avoid the problem of translation into MSA by using Automatic Speech Recognition rather than human scribes. However, these texts were often much shorter than 20-100 words, sometimes only 1 or 2 word utterances; and these short utterances could be common to two or more dialects, with no further indicators for differentiation. Arabic linguistics experts in our team found clear evidence of MSA in numerous dialect texts, possibly introduced by the ASR transcription method; and numerous short utterance instances which had no linguistic evidence of a specific Arabic dialect.

The DSL shared task (Malmasi et al., 2016) was to identify Arabic dialects in texts in five classes: EGY, GLF, LAV, NOR, and MSA; in utterance/phrase level identification which is more challenging than document dialect identification, since short texts have fewer identifying features. Arabic dialects classification is becoming important due to the increasing use of Arabic dialect in social media, and the importance of identification of the dialect before machine translation takes place, or search and retrieval of data (Lu and Mohamed, 2011). Furthermore, identifying the dialect may improve the Part-Of-Speech tagging: for example, the MADAMIRA toolkit identifies the dialect (MSA or EGY) prior to the POS tagging (Pasha et al., 2014). The task of Sentiment Analysis of texts, classifying the text as positive or negative sentiment, is also dialect-specific, as some diagnostic words (especially negation) differ from one dialect to another.

In this paper we describe our method for defining features and choosing the best combination of classifier and feature-set for this task. We show the results of different variants of SMO with different feature-tokenizers. Finally, we conclude the paper by discuss the limitations that affected our results.

2 Related Work

There have been many studies about Arabic dialect Identification. One of these studies presented by Zaidan and Callison-Burch (Zaidan and Callison-Burch, 2014). The authors focused on three Arabic dialects: Levantine, Gulf, and Egyptian and they created a large data set called the Arabic Online Commentary Dataset (AOCD) contained words in all dialects from readers' comments on the three online Arabic newspapers. They obtained 1.24M words from Al-Ghad newspaper (from Jordan to cover Levantine dialect), 18.8M form Al-Riyadh newspaper (from Saudi Arabia to cover Gulf dialect), and 32.1M form Al-Youm Al-Sabe newspaper (from Egypt to cover Egyptian dialect). They classify dialect using Nave Bayes and used wordGram and charcterNGram as features and trained the classifier using unigram, bigram, and trigram models for word, and unigram, trigram, and 5-gram for character model. Based on the dataset they used in training process they found that a unigram word model achieved best accuracy when examine the classifier using 10-fold cross validation (Zaidan and Callison-Burch, 2014). Another study built a system called LAHGA proposed to classify EGY dialect, LEV dialect, and MAG dialect (Lu and Mohamed, 2011). The authors used Tweets as a dataset for training and testing processes. Then begin manually identifying features by reading thousands of tweets and extracting features. They used three different classifiers, which are Nave Bayes classifier, Logistic Regression classifier, and Support Vector Machine classifier. The testing phase was divided into manual testing and cross-validation. During the manual testing process, they removed all noise and chosen 90 tweets, 30 from each dialect, whereas in a 10-flod cross-validation there is not any human intervention. The LAHGA performance shows 90% on a manual test and 75% on cross-validation.

Another research study to classify Arabic dialects used a sentence-level approach to classify whether the sentence was MSA or Egyptian dialect (Elfardy and Diab, 2013). They based the study on a supervised approach using Nave Bayes classifier which was trained on labelled sentences with two types of features: Core Features to indicate if the given sentence is dialectal or non-dialectal. Meta Features to

estimate whether the sentence is informal or not. The system accuracy was about 85.5%.

3 Data

The data for the shared task provided from the DSL Corpus Collection (Ali et al., 2016) is a dataset containing ASR transcripts of utterances by Arabic dialect speakers; there was no guarantee that each utterance was unique to a dialect. The task is performed at the utterance-level and they provided us with two sets. The first set is for training and contains 7,619 utterances labelled and divided unevenly between 5 classes that cover four Arabic dialects (EGY, GLF, LAV, NOR), and MSA (it is not clear how MSA speakers were procured as MSA is not a spoken dialect). Table 1 shows the number of utterances for each class. The second set is for testing, consisting of 1,540 unlabelled utterances. The utterance length ranged from one word to 3305 words with an average of 40 words/utterance and standard deviation = 60. The number of utterances with word count less than 10 words is 1761 = 23.1%. Figure 1 shows the utterances distribution over utterance length.

Classes	Number of utterances
EGY	1578
GLF	1672
LAV	1758
NOR	1612
MSA	999

Table 1: The number of utterances for each class.

Figure 1: The sentences distribution over sentence length.

4 Method

At the beginning we tried to choose the best classifier for the Arabic Dialects Identification (ADI) task from a set of classifiers provided by WEKA (Hall et al., 2009) by measuring the performance of several classifiers on testing with the training dataset, 10-fold cross-validation, and by percentage split which divides the training set into 60% for training and 40% for testing. Table 2 reports results for a range of classifiers that we tried, using the WEKA StringToWordVector filter with WordTokenizer to extract words as features from utterance-strings. SMO was the best performing classifier. Table 3 shows the results of SMO using CharacterNGram Tokenizer with Max=3 and Min=1. Word Tokenizer method, also known as bag of words, is a filter that converts the utterances into a set of attributes that represents the occurrence of words (delimited by space, comma, etc) from the training set. It is designed to keep the n (which we set to 1000) top words per class. NGramWord Tokenizer is similar to Word Tokenizer with the exception that ability to also include word-sequences with the max and min number of words; while CharacterNGram Tokenizer counts 1-2- and/or 3-character n-grams in the utterance-string.

The second column in table 2 shows the results on the same (dialect-labelled) data as those used to train classifier. Third column represents the results on 10-fold cross-validation. The fourth column shows the results on a randomly selected 40% of original training data for test of classifiers trained on the other 60%. After running the experiments in Table 2, we realised that 10-fold cross-validation is very time-consuming (at least 10 times the duration of evaluation on training set or 60:40 percentage split) but produces the same classifier ranking, so we did not repeat the 10-fold cross-validation for table 3.

Classifier	Evaluate on training set	10-fold cross-validation	60% train, 40% test
NaiveBayes	47.09	45.01	43.93
SMO	**89.29**	**52.82**	**50.13**
J48	72.28	43.26	41.5
ZeroR	23.07	23.07	22.41
JRip	35.67	32.76	32.51

Table 2: The accuracy of different classifiers (WordTokenizer).

Classifier	Evaluate on training set	60% train, 40% test
SMO	**94.46**	**53.08**
J48	88.36	37.53
REPTree	53.71	35.56
JRip	41.62	36.35

Table 3: The accuracy of different classifiers (CharacterNGramTokenizer).

Looking at table 2, we noticed that by using SMO we got 6803 utterances correctly classified and 816 utterances misclassified. To improve the identification results we output the misclassified utterances and converted the text from Buckwalter to normal readable Arabic script because looking at the Buckwalter texts is difficult even if you know the Buckwalter transliteration system (Buckwalter, 2002). Then, we asked our Arabic linguistic experts to examine some of the texts which were misclassified, and try to find features which might correctly predict the dialect. Figure 2 shows example of misclassified utterances. The example shows the instance 4 is actually labelled class 2:GLF but the classifier made an error and predicted class 3:NOR.

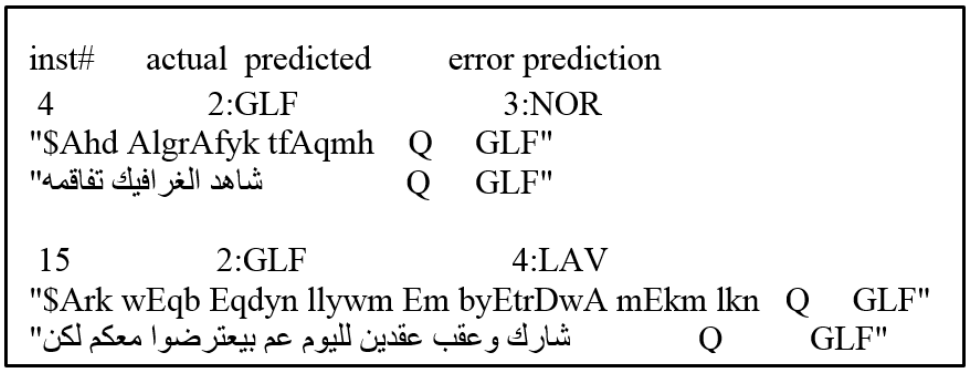

Figure 2: Example of misclassified sentences.

The Arabic linguistics experts analysed the shortcomings in the misclassified utterances from the training data. They found that numerous texts are too short to say anything about their dialect origins, for example: $Ark is a short one-word text which appears unchanged labelled as different dialects. Some of the utterance seem to be entirely MSA despite having dialect labels, possibly due to the Automatic Speech Recognition method used; and a lot of the utterance have at least some MSA in them. Some

utterances that have recognisable dialect words often have words -which are shared between two or more dialects. They even found some utterances labelled as one dialect but evidently containing words not from that dialect; for example in utterance (254) labelled as LAV in the training set contains a non-LAV lexical item, see figure 3.

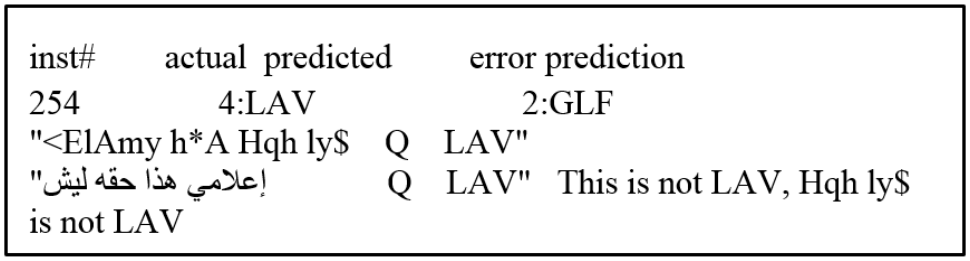

Figure 3: Example of mislabelled sentences.

This analysis led us to conclude that it is impossible in principle for WEKA to classify all instances correctly. There is a proportion of texts that cannot be classified, and this sets a ceiling on accuracy that it is possible to achieve approximate to 90-91%.

4.1 Sequential Minimal Optimization (SMO)

SMO is the WEKA implementation of the Support Vector Machines classifier (SVM) which have been developed for numeric prediction and classifying data by constructing N-dimensional hyper plane to separate data optimally into two categories (Ayodele, 2010). SMV works to find a hypothesis h that reduces the limit between the true error in h will make it on unseen test data and the error on the training data (Joachims, 1998). SMV achieved best performance in text classification task due to the ability of SVM to remove the need for feature selection which means SVM eliminate a high-dimensional feature spaces resulting from the frequent of occurrence of word wi in text. In addition, SVM automatically find good parameter settings (ibid).

4.2 Term Frequency (TF)

Term Frequency represent the frequency of particular word in text (Gebre et al., 2013). Based in our task we found some words usually frequent in one dialect more than other dialects. So we used the weight of TF to indicate the importance of a word in text.

4.3 Inverse Document Frequency (IDF)

Invers Document Frequency tried to scale the weight of frequent words if it appear in different texts (more than one dialects) that is mean a word which appear in many dialect we cannot used as feature (Gebre et al., 2013).

4.4 Features

The first experiments to choose the best classifier to identify Arabic dialects showed that SMO is the best machine learning classifier algorithm, but we may increase accuracy by adjusting parameters and features taken into account.

The WordTokenizer setting assumes features are words or character-strings between spaces while the CharacterNGramTokenizer assumes features are 1/2/3-character sequences. We used the WEKA StringToWordVector filter with WordTokeniser which splits the text into words between delimiters: (full-stop, comma, semi-colon, colon, parenthesis, question, quotation and exclamation mark). After that, we decided to use SMO, but we suggested trying character n-Grams as units, instead of words as units. We used CharacterNGramTokenizer to splits a string into an n-gram with min and max gram. We tried to set Max and Min both to 1 gives a model based on single characters; max and min both to 2 is a char-bigram model; max and min both to 3 gives us a trigram model; max and min to 4 gives a 4-gram model, table 4 shows the results of different gram values when evaluating with the training set and a 60:40 percentage

split of the training set. Table 4 suggests that 4-gram model may be inappropriate as the training data is not sufficiently large.

Features	Evaluate on training set	60% train, 40% test
Character UniGram	43.23	41.11
Character BiGram	78.08	52.4
Character TriGram	**94.62**	49.87
Character FourGram	85.01	50.39

Table 4: The accuracy of SMO classifier with CharcterNGram.

In addition, to improve performance we tried to replace the dimensions of the feature vector with their IDF and TF weight which is a standard method from Information Retrieval (Robertson, 2004). We supposed the models were very similar: (3-1) has all the trigrams of (3-3) and also some bigrams and unigrams but these probably are common to all or most dialects and so do not help in discrimination. However, the Task rules stated that we were restricted to trying our three best classifiers, so at this stage we had to choose three "Best" results. Sometimes the training set score is high, but the 60:40 percentage split score is low; and sometimes the 60:40 percentage split score is high but the Training set score is poor. So, we decided to use 60:40 percentage split as our guide to choose the best combination, because using the training set for training as well as evaluation may over-fit to the training set. Figure 4 below shows the chart that summarises the above four tables for different combinations of TF/IDF and WC values with SMO classifier.

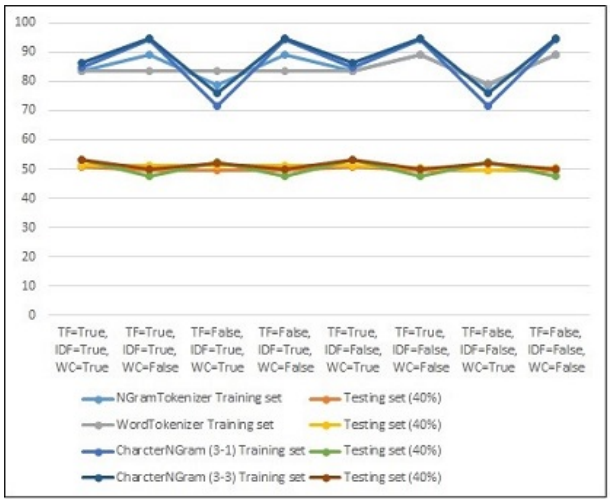

Figure 4: Summary of different combinations of TF/IDF and WC values with SMO classifier.

5 Results

We finally evaluated our system using the supplied separate test data set and submitted three different results using SMO classifier with three different features-sets:

Run1: is obtained by using CharacterNGram, Max=3, Min=3, IDF=True, TF=True, WC=True. We got accuracy around 42%.

Run2: is obtained by using WordTokenizer, IDF=True, TF=True, WC=True, we removed ' delimiter because it is used as a letter in Buckwalter transcription. The performance of this model equals to 37%.

Run3: is obtained by using NGramTokenizer, Max=3, Min=1, IDF=True, TF=True, WC=True, also we removed ' delimiter as in Run2. We got accuracy equals to 38%. Table 5 shows the results of the three runs.

Run	Accuracy	F1 (weighted)
1	42.86	43.49
2	37.92	38.41
3	38.25	38.71

Table 5: The results of the three classifiers.

6 Conclusion

We built systems that classify Arabic dialects in shared task by using WEKA data analytic tool and SMO machine learning algorithm after testing variants of SMO with different tokenizers, IDF, TF, WC values, and comparing results tested on training set (around 80-90% correct) as against using 60% to train and separate 40% for test (around 50% correct). By testing our system on the testing data set we got an average accuracy of 42.85%. We think that we got a low accuracy due to ASR transcription because the most of the misclassified instances are not readily classifiable even by three human Arabic Linguistic experts, which provides strong evidence that a Machine Learning classifier can do no better. Clearly if the training data contains inappropriately-transcribed text and mislabelled instances, this will reduce the ceiling of accuracy that any classifier can achieve. We think that we might combine WordTokenizer and CharacterNGram in the future to improve the results.

References

Ahmed Ali, Najim Dehak, Patrick Cardinal, Sameer Khurana, Sree Harsha Yella, James Glass, Peter Bell and, and Steve Renals. 2016. Automatic dialect detection in arabic broadcast speech. *Interspeech2016*, pages 2934–2938.

Fawzi S Alorifi. 2008. *Automatic Identification of Arabic Dialects Using Hidden Markov Models*. Thesis.

Taiwo Oladipupo Ayodele. 2010. Types of machine learning algorithms.

Fadi Biadsy, Julia Hirschberg, and Nizar Habash. 2009. Spoken arabic dialect identification using phonotactic modeling, 31 March.

Tim Buckwalter. 2002. Arabic transliteration. *URL http://www. qamus. org/transliteration. htm*.

Heba Elfardy and Mona Diab. 2013. Sentence level dialect identification in arabic. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, page 456461.

Binyam Gebrekidan Gebre, Marcos Zampieri, Peter Wittenburg, and Tom Heskes. 2013. Improving native language identification with tf-idfweighting. In *Proceedings of the Eighth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 216–223. Association for Computational Linguistics.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. 2009. The weka data mining software: An update. *SIGKDD Explorations*, 11(1):10–18.

Uri Horesh and William M. Cotter. 2016. Current research on linguistic variation in the arabic-speaking world. *Language and Linguistics Compass*, 10(8):370–381.

Thorsten Joachims. 1998. Text categorization with support vector machines: Learning with many relevant features.

Man Lu and Moustafa Mohamed. 2011. Lahga: Arabic dialect classifier. Report, December 13, 2011.

Shervin Malmasi, Marcos Zampieri, Nikola Ljubei, Preslav Nakov, Ahmed Ali, Jrg Tiedemann, and Liling Tan. 2016. Discriminating between similar languages and arabic dialect identification: A report on the third dsl shared task. In *Proceedings of the 3rd Workshop on Language Technology for Closely Related Languages, Varieties and Dialects (VarDial)*.

Arfath Pasha, Mohamed Al-Badrashiny, Mona Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan M. Roth. 2014. Madamira: A fast, comprehensive tool for morphological analysis and disambiguation of arabic.

Stephen Robertson. 2004. Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60(5):503–520.

Fatiha Sadat, Farnazeh Kazemi, and Atefeh Farzindar. 2014. Automatic identification of arabic language varieties and dialects in social media. In *Proceedings of the Second Workshop on Natural Language Processing for Social Media (SocialNLP)*, pages 22–27.

Omar F. Zaidan and Chris Callison-Burch. 2014. Arabic dialect identification. *Computational Linguistics*, 40(1):171–202.

Marcos Zampieri, Binyam Gebrekidan Gebre, Hernani Costa, and Josef van Genabith. 2015. Comparing approaches to the identification of similar languages. In *Proceedings of the Joint Workshop on Language Technology for Closely Related Languages, Varieties and Dialects*, pages 66–72. Association for Computational Linguistics.